ELSEVIER

Theoretical Computer Science 264 (2001) 127–137

Theoretical Computer Science

www.elsevier.com/locate/tcs

On BPP versus NP ∪ coNP for ordered read-once branching programs

Farid Ablayev[a,*,1], Marek Karpinski[a,b,2], Rustam Mubarakzjanov[c,3]

[a] *Department of Computer Science, University of Bonn, Bonn, Germany*
[b] *International Computer Science Institute, Berkeley, CA, USA*
[c] *Department of Theoretical Cybernetics, University of Kazan 42008, Kazan, Russia*

Accepted April 2000

Abstract

We investigate the relationship between probabilistic and nondeterministic complexity classes PP, BPP, NP and coNP with respect to ordered read-once branching programs (OBDDs). We exhibit two explicit Boolean functions q_n, R_n such that: (1) $q_n : \{0,1\}^n \to \{0,1\}$ belongs to BPP $\setminus$ (NP ∪ coNP) in the context of OBDDs; (2) $R_n : \{0,1\}^n \to \{0,1\}$ belongs to PP $\setminus$ (BPP ∪ NP∪coNP) in the context of OBDDs. Both of these functions are not in AC^0.

1. Preliminaries

Ordered binary decision diagrams (for short OBDDs) are also known as deterministic ordered (or oblivious) read-once branching programs. OBDDs are important tools in the field of digital design and hardware verification (see, for example, [8, 21]). The reason for this is that the manipulation (testing for equivalence and other Boolean operations) with OBDDs can be performed in deterministic polynomial time. But for this convenience we "pay a sensible tax": some important (for practice) Boolean functions cannot be represented by polynomial size OBDDs (see, for example, [16]). So, the important task is to investigate reasonable generalizations of OBDD model of

* Correspondence address. Department of Theoretical Cybernetics, University of Kazan 42008, Kazan, Russia.

E-mail addresses: ablayev@ksu.ru (F. Ablayev), marek@cs.uni-bonn.de (M. Karpinski), rustam.mubarakzjanov@ksu.ru (R. Mubarakzjanov).

[1] Partially supported by the Volkswagen-Stiftung and Russia Fund for Basic Research 96-01-01692.

[2] Partially supported by DFG Grant KA 673/4-1, by the ESPRIT BR Grants 7097, and EC-US 030, by the Volkswagen–Stiftung and by the Max-Planck Research Prize.

[3] Supported by the Russia Fund for Basic Research 96-01-01692.

0304-3975/01/$ - see front matter
PII: S0304-3975(00)00216-4

computation and to compare complexity classes defined, with respect to these general models.

We consider randomized (with constant error of computation) and probabilistic (with unrestricted error of computation) models of OBDD in this paper. We investigate relationships between complexity classes PP, BPP, NP, and coNP based on OBDD model of computation and compare them with another known complexity class AC^0. Recall that AC^0 is the class of Boolean functions computable by polynomial size unbounded fanin circuits of constant depth (cf., [6]). In [11] the complexity classes NP and coNP for read-once branching programs are compared with the class AC^0.

We recall some basic definitions [17].

A *deterministic* branching program P is a directed acyclic multi-graph with a source node and two distinguished sink nodes: accepting and rejecting. The outdegree of each nonsink (internal) node is exactly 2 and the two outgoing edges are labeled by $x_i = 0$ and $x_i = 1$ for a variable x_i associated with the node. Call such a node an x_i-node. The label "$x_i = \delta$" indicates that only inputs satisfying $x_i = \delta$ may follow this edge in a computation. A branching program P computes a Boolean function $h_n : \{0,1\}^n \to \{0,1\}$ in the obvious way: for each $\bar{\sigma} \in \{0,1\}^n$ we let $h_n(\bar{\sigma}) = 1$ iff there is a directed path starting in the source and leading to the accepting node such that all labels $x_i = \sigma_i$ along this path are consistent with $\bar{\sigma} = \sigma_1\sigma_2 \cdots \sigma_n$.

A branching program becomes *nondeterministic* if we allow "guessing nodes" that is, nodes with two outgoing edges being unlabeled. A nondeterministic branching program P computes a function h_n in an obvious way; that is, $h_n(\bar{\sigma}) = 1$ iff there exists (at least one) computation on $\bar{\sigma}$ starting in the source node and leading to the accepting node.

A *probabilistic* branching program has, in addition to its standard (deterministic) nodes, specially designated nodes called random ("coin-toss") nodes. Each such node corresponds to a random input y_i having random values from $\{0,1\}$. An output of such a program is a random variable.

We say that a probabilistic branching program p-computes, $p \in (0,1]$, a function h if it outputs 1 with a probability at least p for an input $\bar{\sigma}$ if $h(\bar{\sigma}) = 1$ and outputs 1 with probability less than p if $h(\bar{\sigma}) = 0$. We say that a probabilistic branching program (a,b)-computes a function h if it outputs 1 with probability at least b for an input $\bar{\sigma}$ such that $h(\bar{\sigma}) = 1$ and it outputs 1 with probability at most a for an input $\bar{\sigma}$ such that $h(\bar{\sigma}) = 0$. We call probabilistic branching program a randomized branching program if it $(1/2 - \varepsilon, 1/2 + \varepsilon)$-computes some function h for positive constant $\varepsilon \in (0, 1/2)$.

The size or complexity of a deterministic (*nondeterministic*) *branching program is the number of its internal nodes* (*internal nodes without guessing nodes*). *The size of a probabilistic branching program is the sum of numbers of its internal and random nodes.*

Since branching programs are a nonuniform model of computation, asymptotic statements about the size refer to the families of branching programs containing one program for each input size.

A read-once branching program is a branching program in which for arbitrary path each variable is tested not more than once. An ordered read-once branching program is

a read-once branching program which respects certain fixed ordering π of the variables, i.e., if an edge leads from an x_i-node to an x_j-node the condition $\pi(i) < \pi(j)$ has to be fulfilled. In the area of circuits verification, the ordered read-once branching programs are also known as OBDDs.

Following the notations of [18], we denote the class of Boolean functions computable by polynomial size nondeterministic branching programs by NP−BP. The class coNP−BP contains all Boolean functions with the negations computable by polynomial size nondeterministic branching programs.

Let PP−BP be the class of functions (more formally the class of sequences of functions) which are 1/2-computable by polynomial size probabilistic branching programs.

Let $\mathrm{BPP}_{\varepsilon} - \mathrm{BP}$ be the class of functions (more formally, the class of sequences of functions) which are $(1/2 - \varepsilon, 1/2 + \varepsilon)$-computable by polynomial size probabilistic branching programs. Furthermore, let

$$\mathrm{BPP} - \mathrm{BP} := \bigcup_{0 < \varepsilon \leqslant 1/2} \mathrm{BPP}_{\varepsilon}\text{-BP}.$$

We define analogous classes based on OBDD model of computation using "OBDD" as suffixes.

Using the fact that BPP = coBPP and PP = coPP, we study 4 complexity classes: NP, coNP, BPP, PP based on OBDD model of computation. What is the relationship between these classes? It is also interesting to compare these classes with the class AC^0.

In 1996, Ablayev and Karpinski [2] found a function f_n which belongs to BPP−OBDD (and at the same time to coNP−OBDD) but did not belong to NP−OBDD. In 1997, Ablayev found a function in the class $\mathrm{NP} - \mathrm{OBDD} \setminus \mathrm{BPP} - \mathrm{OBDD}$. These results are valid for complexity classes based on ordered branching programs. In 1997, Sauerhoff [18] showed that permutation function PERM (see [14, 10] for lower bound in nondeterministic case) is in $(\mathrm{BPP} - \mathrm{OBDD} \cap \mathrm{coNP} - \mathrm{OBDD}) \setminus \mathrm{NBP} - \mathrm{BP1}$ (BP1 stands for *read-once branching programs*). For an overview of known upper and lower bounds on randomized OBDDs and read-k-times branching programs see [12].

We present function $q_n : \{0,1\}^n \to \{0,1\}$, belonging to $\mathrm{BPP} \setminus (\mathrm{NP} \cup \mathrm{coNP})$ in the context of OBDDs. The paper [5] presented an explicit Boolean function $r_n : \{0,1\}^n \to \{0,1\}$, in $\mathrm{PP} \setminus (\mathrm{BPP} \cup \mathrm{NP})$ in the context of OBDDs. Using another technique [13], we present an explicit Boolean function $R_n : \{0,1\}^n \to \{0,1\}$ that belongs to $\mathrm{PP} \setminus (\mathrm{BPP} \cup \mathrm{NP} \cup \mathrm{coNP})$ in the context of OBDDs.

2. Probabilistic branching programs

We consider general probabilistic branching programs in this section. We show that any reasonable definitions of probabilistic complexity classes do not increase complexity class PP−BP. Our constructions do not explore the technique of multi-readings of variables. The last will be important in the following.

Definition 1. Let $\{p_n\}$ be a sequence of numbers in $(0,1)$. We say that function (more formally: sequence of functions) h_n belongs to a complexity class $\mathrm{PP}_{\{p_n\}}-\mathrm{BP}$ iff for any natural number n there is a polynomial size probabilistic branching program B_n with n deterministic inputs which p_n-computes the function h_n of n variables.

We denote $\mathrm{PP}_{\{p_n\}}-\mathrm{BP}$ by $\mathrm{PP}_p-\mathrm{BP}$ if $p_n=p$ for any n.
The following property is obvious

Property 1.

$$\mathrm{PP}_1-\mathrm{BP}=\mathrm{coNP}-\mathrm{BP}, \qquad \mathrm{NP}-\mathrm{BP}=\mathrm{coPP}_1-\mathrm{BP}.$$

Lemma 1. *For arbitrary* p, $0<p\leqslant 1$ *it is true that*

$$\mathrm{PP}_p-\mathrm{BP}\subseteq \mathrm{PP}\text{-}\mathrm{BP}.$$

Proof. Let a function h_n be in $\mathrm{PP}_p-\mathrm{BP}$. We construct a probabilistic branching program B_n^2 which $1/2$-computes h_n. For any natural number n there is a probabilistic branching program B_n which p-computes h_n. Let $\bar{\sigma}$ be an input sequence such that $f_n(\bar{\sigma})=1$ and the probability $p(\bar{\sigma})$ of accepting $\bar{\sigma}$ by B_n is $\min\{p(\bar{\alpha})\,|\,h_n(\bar{\alpha})=1\}$. Then $p(\bar{\sigma})=p'\geqslant p$. The input sequence $\bar{\sigma}$ gives in a natural way an "only-random" branching program $B_n(\bar{\sigma})$ with the probability of leading accepting node p'. Denote by $B_n'(\bar{\sigma})$ a branching program $B_n(\bar{\sigma})$ where accepting (rejecting) nodes are replaced by rejecting (accepting) nodes.

B_n^2 is the following probabilistic branching program. The source node corresponds to a random input y_0. Two arcs labeled by "$y_0=0$" and "$y_0=1$" follow from the source to $B_n'(\bar{\sigma})$ and B_n. The probability function $p_1(\mathbf{x})$ of leading accepting node for B_n^2 has the following properties.

For an input sequence $\bar{\alpha}$ such that $f_n(\bar{\alpha})=1$, $p_1(\bar{\alpha})=1/2(1-p')+1/2\,p(\bar{\alpha})=1/2(1-p'+p(\bar{\alpha}))\geqslant 1/2$.

For an input sequence $\bar{\alpha}$ such that $f_n(\bar{\alpha})=0$, $p_1(\bar{\alpha})=1/2(1-p')+1/2\,p(\bar{\alpha})<1/2(1-p'+p')=1/2$. □

Theorem 1. *For any sequence of numbers* $\{p_n|(1/2)^{poly(n)}\leqslant p_n\leqslant 1-(1/2)^{poly(n)}\}$ *it holds that*

$$\mathrm{PP}_{\{p_n\}}-\mathrm{BP}=\mathrm{PP}-\mathrm{BP}.$$

Proof. It is enough to show that for arbitrary natural n, for arbitrary function $\{f_n\}\in$ PP-BP, there is a polynomial size probabilistic branching program B_n which p_n-computes f_n. Denote by B_n' a probabilistic branching program which $1/2$-computes f_n. Such a branching program exists by Lemma 1. Denote by $p(x)$ the accepting probability of input x and call it for short a probability function.

Let ε_n be a number such that $1/2-\varepsilon_n=\max\{p(\bar{\sigma})|f_n(\bar{\sigma})=0,|\bar{\sigma}|=n\}$. Obviously, $\varepsilon_n\geqslant(1/2)^{poly(n)}$. We have to investigate two possibilities: $p_n<1/2$ and $p_n>1/2$. For

both these cases, we take an "only-random" branching program B'_n where the probability of leading accepting node is p'_n. For the first case, $2p_n \leqslant p'_n < 2p_n/(1-2\varepsilon_n)$, for the second one, $2p_n - 1 \leqslant p'_n < (2p_n - 1 + 2\varepsilon_n)/(1+2\varepsilon_n)$.

B_n^2 is a probabilistic branching program consisting of two parts. The first part of B_n^2 is the branching program B'_n. The second part is a probabilistic branching program B_n: its source node is identified with the accepting node of B'_n for $p_n < 1/2$ and with the rejecting node for $p_n > 1/2$. The probabilistic branching program B_n^2 p_n-computes f_n.

Indeed, if $p_1(\mathbf{x})$ is the probability function of B_2^n then,

1. if $p_n < 1/2$,
   (a) for an input sequence $\bar{\sigma}$ such that $f_n(\bar{\sigma}) = 1$, $p_1(\bar{\sigma}) = p'_n p(\bar{\sigma}) \geqslant 1/2 p'_n \geqslant p_n$;
   (b) for an input sequence $\bar{\sigma}$ such that $f_n(\bar{\sigma}) = 0$, $p_1(\bar{\sigma}) \leqslant p'_n(1/2 - \varepsilon_n) < p_n$;
2. if $p_n > 1/2$,
   (a) for an input sequence $\bar{\sigma}$ such that $f_n(\bar{\sigma}) = 1$, $p_1(\bar{\sigma}) = p'_n + (1-p'_n)p(\bar{\sigma}) \geqslant 1/2 + 1/2 p'_n \geqslant p_n$;
   (b) for an input sequence $\bar{\sigma}$ such that $f_n(\bar{\sigma}) = 0$, $p_1(\bar{\sigma}) \leqslant p' + (1-p')(1/2 - \varepsilon_n) < p_n$.

□

Clearly, we have that if guessing nodes of nondeterministic branching programs are replaced by random ones, one obtains a probabilistic branching program p_n-computing the same function for some p_n. Therefore the following is true.

Corollary 1. $\mathrm{NP} - \mathrm{BP} \subseteq \mathrm{PP} - \mathrm{BP}$.

3. Functions and results

Recall that the results of the previous section do not depend on the number of input readings. Therefore all these results are valid for OBDDs. Thus we can state the following.

Property 2. $\mathrm{NP} - \mathrm{OBDD} \subseteq \mathrm{PP} - \mathrm{OBDD}$.

Firstly, we exhibit an explicit Boolean function $q_n : \{0,1\}^n \to \{0,1\}$ such that 1) q_n is *easy* for randomized OBDD (ROBDD for short) and 2) q_n and its negation are *hard* for nondeterministic OBDD. We use the function f_n from [3] for construction of q_n. The Boolean function f_n of $n = 4l$ variables is specified as follows. We say that even bit x_i, $i \in \{2,4,\ldots,4l\}$, has type 0 (1) if the corresponding odd bit x_{i-1} is 0 (1). For a sequence $\bar{\sigma} \in \{0,1\}^{4l}$, denote by $\bar{\sigma}^0$ ($\bar{\sigma}^1$) the subsequence of $\bar{\sigma}$ that consists of all even bits of type 0 (1).

The function $f_n : \{0,1\}^n \to \{0,1\}$ is defined as follows: $f_n(\bar{\sigma}) = 1$ *iff* $\bar{\sigma}^0 = \bar{\sigma}^1$.

Let $l \geqslant 1$, $n = 4l$. We define the Boolean function q_{2n} of $2n$ variables as follows:

$$q_{2n}(x_1,\ldots,x_{2n}) = f_n(x_1,\ldots,x_n) \quad \& \quad \neg f_n(x_{n+1},\ldots,x_{2n}).$$

Theorem 2. *For $n=4l$, $\varepsilon(n)\in(0,1/2)$, the function q_{2n} is $(\varepsilon(n),1-\varepsilon(n))$-computable by an ROBDD of size*

$$O\left(\frac{n^6}{\varepsilon^3(n)}\log^2\frac{n}{\varepsilon(n)}\right).$$

Any nondeterministic OBDD that computes the function q_{2n} or the function $\neg q_{2n}$ has a size of at least 2^l.

Proof. It is shown in [3] that the function f_n can be $(\varepsilon(n),1)$-computed by a randomized read-once ordered branching program of size

$$O\left(\frac{n^6}{\varepsilon^3(n)}\log^2\frac{n}{\varepsilon(n)}\right).$$

The same construction as in [3] can be used for branching program B that computes q_{2n}. The first part of B is a randomized branching program B_1 that $(\varepsilon',1)$-computes the function $f_n(x_1,\ldots,x_n)$. Then, the accepting sink node of B_1 is identified with a source node of a branching program B_2 that $(\varepsilon'',1)$-computes $f_n(x_{n+1},\ldots,x_{2n})$. Finally, we change the places of the sink nodes of B_2.

The program B outputs 1 with probability at most ε' for an input $\bar\sigma$ such that $q_{2n}(\bar\sigma)=0$. The error can occur only for $\bar\sigma$ such that $f_n(\sigma_1,\ldots,\sigma_n)=0$ and $f_n(\sigma_{n+1},\ldots,\sigma_{2n})=0$.

The program B outputs 1 with probability at least $1-\varepsilon''$ for an input $\bar\sigma$ such that $q_{2n}(\bar\sigma)=1$.

If $\varepsilon'=\varepsilon''=\varepsilon(n)$ then B is an ROBDD, as required.

It follows from [3] that any nondeterministic ordered read-once branching program that computes the function f_n, $n=4l$, has a size of at least 2^{l-1}.

We give here a simpler proof than in [3] that nondeterministic ordered read-once branching program B' computing f_{4l} has a size of at least 2^l. We shall use this construction also later. Let B' have an ordering τ of variables. For ordering τ denote by $\tau^0=\{i_1,i_2,\ldots,i_l\}$ a subsequence of τ that consists of the first l even numbers of τ. Similarly, denote by $\tau^1=\{j_1,j_2,\ldots,j_l\}$ a subsequence of τ that consists of the last l even numbers of τ.

Call a sequence $\bar\sigma\in f_n^{-1}(1)$ τ-hard if all its even bits σ_i, $i\in\tau^0$, are of "type" 0 and all its even bits σ_j, $j\in\tau^1$, are of "type" 1. Denote

$$X^\tau=\{\bar\sigma\in\{0,1\}^{4l}:\ \bar\sigma \text{ is } \tau\text{-hard}\}.$$

The cardinality of X^τ is equal to 2^l. Let Q be a set of nodes of B' in a case where exactly l even bits are read by B'. Every sequence of X^τ corresponds to at least one node of Q and different sequences correspond to different nodes. Therefore the cardinality of Q is not less than 2^l.

Obviously, $q_{2n}(x_1,\ldots,x_n,1,1,\ldots,1)=f_n(x_1,\ldots,x_n)$. If $f_n(\sigma_1,\ldots,\sigma_n)=1$ then $\neg q_{2n}(\sigma_1,\ldots,\sigma_n,x_{n+1},\ldots,x_{2n})=f_n(x_{n+1},\ldots,x_{2n})$. □

Corollary 2. $q_{2n} \in \mathrm{BPP} - \mathrm{OBDD} \backslash (\mathrm{NP} - \mathrm{OBDD} \cup \mathrm{coNP} - \mathrm{OBDD})$.

We exhibit now an explicit Boolean function $r_n : \{0,1\}^n \to \{0,1\}$, which can be computed by polynomial size probabilistic OBDD but, which is *hard* for nondeterministic and randomized OBDDs. We use for the construction of r_n the function f_n from [3] and the function g_n from [1, 19]. Let n be an integer and let $p[n]$ be the smallest prime greater than or equal to n. Then, for every integer s, let $\omega_n(s)$ be defined as follows. Let j be the unique integer satisfying $j = s \bmod p[n]$ and $1 \leqslant j \leqslant p[n]$. Then, $\omega_n(s) = j$, if $1 \leqslant j \leqslant n$, and $\omega_n(s) = 1$ otherwise.

For every n, the Boolean function $g_n : \{0,1\}^n \to \{0,1\}$ is defined as $g_n(\bar{\sigma}) = \sigma_j$, where $j = \omega_n(\sum_{i=1}^{n} i\sigma_i)$.

It is shown in [1] that the function g_n is in $\mathrm{NP} - \mathrm{OBDD} \backslash \mathrm{BPP} - \mathrm{OBDD}$.

Let $l \geqslant 1$, $n = 4l$. Define a Boolean function r_n of n variables as follows:

$$r_{4l}(\sigma_1, \ldots, \sigma_{4l}) = f_{4l}(\sigma_1, \ldots, \sigma_{4l}) \quad \& \quad g_l(\bar{\sigma}^0).$$

Lemma 2 (Ablayev et al. [5]). $r_n \in \mathrm{PP} - \mathrm{OBDD} \backslash (\mathrm{BPP} - \mathrm{OBDD} \cup \mathrm{NP} - \mathrm{OBDD})$.

Proof. The probabilistic OBDD B computes r_{4l} as follows. It starts with the probability $1/2$, a probabilistic OBDD B_1, and it starts with probability $1/2$, a probabilistic OBDD B_2.

Because of Property 2, and the construction of a nondeterministic branching program computing g_l, there is a probabilistic OBDD B_1 which $1/2$-computes g_l, and reads the variables in the prescribed order $(1, 2, \ldots, n)$. An ROBDD B_2 which $(\varepsilon, 1)$-computes the function f_n reads the variables in the prescribed order too, $\varepsilon < 1/2$.

The following proves that the OBDD B probabilistically $3/4$-computes the function r_{4l}.

If for an input $\bar{\sigma}$ the function $r_{4l}(\sigma_1, \ldots, \sigma_{4l}) = 1$ then $f_{4l}(\sigma_1, \ldots, \sigma_{4l}) = g_l(\bar{\sigma}^0) = 1$. The OBDD B computes 1 with a probability of at least

$$1/2 \cdot 1 + 1/2 \cdot 1/2 = 3/4.$$

If for an input $\bar{\sigma}$ the function $r_{4l}(\sigma_1, \ldots, \sigma_{4l}) = 0$ then there are three possibilities:

1. $f_{4l}(\sigma_1, \ldots, \sigma_{4l}) = 0$, $g_l(\bar{\sigma}^0) = 1$. Then the OBDD B computes 1 with probability in most

$$1/2 \cdot \varepsilon + 1/2 \cdot 1 < 3/4.$$

2. $f_{4l}(\sigma_1, \ldots, \sigma_{4l}) = 1$, $g_l(\bar{\sigma}^0) = 0$. Then the OBDD B computes 1 with probability less than

$$1/2 \cdot 1 + 1/2 \cdot 1/2 = 3/4;$$

3. $f_{4l}(\sigma_1, \ldots, \sigma_{4l}) = 0$, $g_l(\bar{\sigma}^0) = 0$. Then the OBDD B computes 1 with probability less than

$$1/2 \cdot \varepsilon + 1/2 \cdot 1/2 < 1/2.$$

The function r_n is in PP − OBDD. Because the function g_n does not belong to BPP−OBDD the function r_{4l} does not belong to BPP−OBDD either. Indeed, if for $i=1,\ldots,l$,

1. $\sigma_{4i-3}=0$,
2. $\sigma_{4i-1}=1$,
3. $\sigma_{4i-2}=\sigma_{4i}$,

then $r_{4l}(\sigma_1,\ldots,\sigma_{4l})=g_l(\sigma_2,\sigma_6,\ldots,\sigma_{4i-2},\ldots,\sigma_{4l-2})$.

To show that the function r_{4l} does not belong to NP−OBDD we use the set

$$Y^\tau=\{\bar\sigma\in\{0,1\}^{4l}:\ \bar\sigma \text{ is } \tau\text{-hard and } g_l(\bar\sigma^0)=1\}$$

in the construction in the proof of Theorem 1, instead of

$$X^\tau=\{\bar\sigma\in\{0,1\}^{4l}:\ \bar\sigma \text{ is } \tau\text{-hard}\}.$$

Analogously to the idea of the proof of Theorem 1, the size of nondeterministic OBDD computing r_{4l} is not less than the cardinality of Y^τ.

To evaluate the cardinality of Y^τ we use the method of [1].

We use the following result (see [9, 20]).

Lemma 3. *For every n large enough, if $p(n)$ is the smallest prime greater than or equal to n, then the following is true. If $A\subseteq\{0,1,2,\ldots,p(n)-1\}$ and $|A|\geqslant 3\sqrt{n}$, then for every t, $0\leqslant t\leqslant p(n)-1$, there is a subset $B\subseteq A$ such that the sum of the elements of B is equal to t.*

Let $m=\lceil 3\sqrt{l}\rceil$. For any $\bar\alpha\in\{0,1\}^{l-m}$ there is a $\bar\beta\in\{0,1\}^m$ such that $g_l(\bar\alpha,\bar\beta)=1$.

Indeed, if $\bar\alpha=\mathbf{0}$ then $\bar\beta=\mathbf{0}$.

If there is a t such that $\alpha_t=1$ and $\sum_{i=1}^{l-m} i\alpha_i=s$ then because of Lemma 1 there is a $\bar\beta\in\{0,1\}^m$ such that for $\bar\sigma=(\bar\alpha,\bar\beta)$, $\omega_n(\sum_{j=l-m+1}^{l} j\sigma_j+s)=t$. Therefore $g_l(\bar\sigma)=1$.

Thus $|Y^\tau|\geqslant|\{\bar\alpha:\ \bar\alpha\in\{0,1\}^{l-m}\}|=2^{l-\lceil 3\sqrt{l}\rceil}$.

Define a Boolean function R_{2n} of $2n$ variables as follows:

$$R_{2n}(\mathbf{x})=R_{2n}(\mathbf{x}_1,\mathbf{x}_2)=r_n(\mathbf{x}_1)+r_n(\mathbf{x}_2).$$

Theorem 3. $R_{2n}\in \mathrm{PP}-\mathrm{OBDD}\backslash(\mathrm{BPP}-\mathrm{OBDD}\cup\mathrm{NP}-\mathrm{OBDD}\cup\mathrm{coNP}-\mathrm{OBDD})$.

Proof. A branching program $B(R_{2n})$ computing R_{2n} consists of two parts. The first part of $B(R_{2n})$ is a randomized branching program B_1 that computes the function $r_n(\mathbf{x}_1)$. Then the rejecting sink node of B_1 is identified with source node of branching program B_2 that computes $r_n(\mathbf{x}_2)$. The accepting sink node of B_1 is identified with the source node of branching program B_2' that is a copy of B_2 with one exception: the places of the sink nodes are changed.

Let $\mathbf{x}=(\mathbf{x}_1,\mathbf{x}_2)$. If the probability of computing 1 on $\mathbf{x}_i$ by B_i is p_i for $i=1,2$, then $B(R_{2n})$ computes 1 with probability

$$p = p_1 + p_2 - 2p_1p_2 = p_1(1-2p_2) + p_2.$$

Let $R_{2n}(\mathbf{x})=1$. Then $r_n(\mathbf{x}_1)=1$, $r_n(\mathbf{x}_2)=0$ or $r_n(\mathbf{x}_1)=0$, $r_n(\mathbf{x}_2)=1$. In the first case

$$p_1 \geqslant 1/2, \qquad p_2 < 1/2.$$

Therefore, $1-2p_2>0$ and

$$p = p_1(1-2p_2) + p_2 \geqslant 1/2(1-2p_2) + p_2 = 1/2.$$

If $r_n(\mathbf{x}_1)=0$ and $r_n(\mathbf{x}_2)=1$ then $p\geqslant 1/2$ too.

Let $R_{2n}(\mathbf{x})=0$. Then $r_n(\mathbf{x}_1)=r_n(\mathbf{x}_2)=0$ or $r_n(\mathbf{x}_1)=r_n(\mathbf{x}_2)=1$. In the first case

$$p_1 < 1/2, \qquad p_2 < 1/2.$$

Therefore,

$$p = p_1(1-2p_2) + p_2 < 1/2(1-2p_2) + p_2 = 1/2.$$

If $r_n(\mathbf{x}_1)=r_n(\mathbf{x}_2)=1$, then

$$p_1 \geqslant 1/2, \qquad p_2 \geqslant 1/2.$$

Therefore $1-2p_2 \leqslant 0$ and

$$p = p_1(1-2p_2) + p_2 \leqslant 1/2(1-2p_2) + p_2 = 1/2.$$

Therefore, $B(R_{2n})$ is a probabilistic branching program that 1/2-computes the function R_{2n}. □

Using the permutation function PERM instead of f_n we can prove the following.

Theorem 4. *There are explicit Boolean functions that belong to the following complexity classes:*

1. $\mathrm{BPP}-\mathrm{OBDD}\backslash(\mathrm{NP}-\mathrm{BP1}\cup\mathrm{coNP}-\mathrm{BP1})$,
2. $\mathrm{PP}-\mathrm{OBDD}\backslash(\mathrm{BPP}-\mathrm{OBDD}\cup\mathrm{NP}-\mathrm{BP1}\cup\mathrm{coNP}-\mathrm{BP1})$.

In conclusion, we prove that the functions q_n, r_n, R_n do not belong to AC^0.

Property 3 (Ablayev & Karpinski [3]). $f_n \notin \mathrm{AC}^0$.

Proof. To prove that $f_n \notin \mathrm{AC}^0$ it is enough to show that $\mathrm{PARITY}(x_1,x_2,\ldots,x_{2l})$ is AC^0-reducible to the function $f_{n'}$ for some n'.

Let $n=4l$. Denote by f_n^t, $0\leqslant t\leqslant n/2=2l$, a subfunction of the function $f_{n+|n-4t|}$ obtained by setting all even input bits of $f_{n+|n-4t|}$ to 0, and the last $|n/2-2t|$ odd

input bits to 1, if $n \geqslant 4t$, and otherwise to 0. Obviously, if the rest of the odd bits form a sequence $\{\sigma_1, \sigma_2, \ldots, \sigma_{2l}\}$ then

$$f_n^t(\sigma_1, \sigma_2, \ldots, \sigma_{2l}) = 1,$$

if and only if this sequence contains exactly t bits equal to 1. Therefore

$$\mathrm{PARITY}(x_1, x_2, \ldots, x_{2l}) = \bigvee_{s=1}^{l} f_{4l}^{2s}(x_1, x_2, \ldots, x_{2l}). \qquad \square$$

Corollary 3. $q_{2n} \notin \mathrm{AC}^0$.

Proof. Indeed $q_{2n}(x_1, \ldots, x_n, 1, 1, \ldots, 1) = f_n(x_1, \ldots, x_n)$. $\square$

Corollary 4. $r_{4l} \notin \mathrm{AC}^0$.

Proof. Use in the construction of the function $f_{4l}^{2s}(x_1, x_2, \ldots, x_{2l})$ (Proof of Proposition 3), the function $r_{4l+|4l-8s|}$ instead of $f_{4l+|4l-8s|}$. $\square$

Corollary 5. $R_{8l} \notin \mathrm{AC}^0$.

Acknowledgements

We would like to thank Stephen Ponzio, Sasha Razborov, and Thomas Tierauf for helpful discussions on the subject of this paper.

References

[1] F. Ablayev, Randomization and Nondeterminism are Incomparable for ordered read-once branching programs, Proc. ICALP'97, Lecture Notes in Computer Science, Vol. 1256, Springer, Berlin, 1997, pp. 195–202; ECCC TR97-021, 1997, available at http://www.eccc.uni-trier.de/eccc/.

[2] F. Ablayev, M. Karpinski, On the power of randomized branching programs, Proc. ICALP'96, Lecture Notes in Computer Science, Vol. 1099, Springer, Berlin, 1996, pp. 348–356.

[3] F. Ablayev, M. Karpinski, On the power of randomized ordered branching programs, ECCC TR98-004, 1998, available at http://www.eccc.uni-trier.de/eccc/.

[4] F. Ablayev, M. Karpinski, A lower bound for integer multiplication on randomized read-once branching programs, ECCC TR98-011, 1998, available at http://www.eccc.uni-trier.de/eccc/.

[5] F. Ablayev, M. Karpinski, R. Mubarakzjanov, On BPP versus NP∪coNP for ordered read-once branching programs, Proc. Randomized Algorithms 1998, Bruno, 1998.

[6] R. Boppana, M. Sipser, The Complexity of Finite Functions, in Handbook of Theoretical Computer Science, J. Van Leeuwen (Ed.), Vol A. Algorithms and Complexity, MIT Press and Elsevier, The Netherlands, 1990, pp. 757–804.

[7] A. Borodin, A. Razborov, R. Smolensky, On lower bounds for read-k-times branching programs, Comput. Complexity 3 (1993) 1–18.

[8] R. Bryant, Symbolic boolean minipulation with ordered binary decision diagrams, ACM Comput. Surveys 24 (3) (1992) 293–318.

[9] J. Dias da Silva, Y. Hamidoune, Cyclic spaces for Grassmann derivatives and additive theory, Bull. London Math. Soc. 26 (1994) 140–146.

[10] S. Jukna, On the effect of null-chains on the complexity of contact schemes, Proc. of FCT, Lecture Notes in Computer Science, Vol. 380, 1989, pp. 246–256.
[11] S. Jukna, A. Razborov, P. Savicky, I. Wegener, On P versus $\mathrm{NP} \cap \mathrm{co} - \mathrm{NP}$ for decision trees and read-once branching programs, ECCC TR97-023, 1997, available at http://www.ecc.uni-trier.de/eccc/.
[12] M. Karpinski, On the computational power of randomized branching programs, Proc. Randomized Algorithms 1998, Bruno, 1998.
[13] M. Karpinski, R. Mubarakzjanov, Some separation problems on randomized OBDDs, University of Bonn, 85196-CS, 1998 available http://cs.uni-bonn.de/info5/publications/cs-1998-ln.html.
[14] M. Krause, C. Meinel, S. Waack, Separating the eraser turing machine classes $\mathrm{L}_e, \mathrm{NL}_e$, $\mathrm{co} - \mathrm{NL}_e$ and P_e, Proc. of MFCS, Lecture Notes in Computer Science, Vol. 324, pp. 405–413.
[15] W. Masek, A fast algorithm for the string editing problem and decision graph complexity, M.Sc. Thesis, Massachusetts Institute of Technology, Cambridge, May 1976.
[16] S. Ponzio, A lower bound for integer multiplication with read-once branching programs, Proc. 27th ACM STOC, 1995, pp. 130–139.
[17] A. Razborov, Lower bounds for deterministic and nondeterministic branching programs, Proc. FCT'91, Lecture Notes in Computer Science, Vol. 529, Springer, Berlin, 1991, pp. 47–60.
[18] M. Sauerhoff, A lower bound for randomized read-k-times branching programs, ECCC, TR97-019, 1997, available at http://www.eccc.uni-trier.de/eccc/.
[19] P. Savicky, S. Zak, A large lower bound for 1-branching programs, ECCC, Revision 01 of TR96-036, 1996, available at http://www.eccc.uni-trier.de/eccc/.
[20] P. Savicky, S. Zak, A hierarchy for $(1, +k)$-branching programs with respect to k, ECCC, TR96-050, 1996, available at http://www.eccc.uni-trier.de/eccc/.
[21] I. Wegener, Efficient data structures for boolean functions, Discrete Math. 136 (1994) 347–372.